MANUALE DEL BASIC

SELCOM®
elettronica

DIVISIONE NUOVE TECNOLOGIE 48100 RAVENNA Italy Via Lametta, 9 tel. 0544/35365 (2 Linee)

INDICE

INDICE DEI COMANDI BASIC

INTRODUZIONE

Congratulazioni e benvenuti nell'affascinante mondo dei personal computers. Le possibilità del Vostro LEMON II sono illimitate per cui lo scopo di questo manuale di riferimento è quello di introdurvi alla logica della programmazione e al metodo LEMON di usare i programmi.

Il linguaggio che sarà usato per comunicare con il LEMON II è il BASIC.

Questo linguaggio è stato sviluppato nel Dartmouth College di Hanover (Ohio) USA, alla fine degli anni sessanta e da allora è divenuto uno dei più importanti linguaggi di programmazione il cui nome deriva da «Beginner All - purpose Symbolic Instruction Code», cioè (Codice simbolico di istruzioni per principianti).

Per questo può dare l'impressione di essere un linguaggio di programmazione molto semplificato e di ridotta potenza operativa; in verità vale il contrario: il BASIC può essere un linguaggio semplice, ma molto potente.

Precisamente il Vostro LEMON II usa come linguaggio il BASIC che è una ESTENSIONE del diffusissimo «INTERPRETE BASIC STANDARD» a virgola mobile della Microsoft, residente in ROM di 10K bytes. In ROM di 2K si trova il Monitor che è del tipo autostart, cioè il sistema all'accensione è immediatamente operativo in BASIC.

CARATTERISTICHE DEL LEMON II

HARAWARE

Microprocessore: 65Ø2 a 1,Ø23 MHZ
Memoria RAM: 48 Kbyte
Memoria video: 1 Kbyte
Memoria ROM: 1Ø Kbyte occupati dal BASIC ESTESO
2 Kbyte occupati dal Monitor
4 Kbyte sulle interfacce

Interfaccia per unità e cassetta
Porte di I/O TTL
Connettori sul bus di sistema: 8

DISPLAY VIDEO

TESTO
- Visualizzazione di 96Ø caratteri (4Ø caratteri per riga, 24 righe).
- Caratteri maiuscoli generati da una matrice di punti 5x7.
- Completo controllo del cursore.
- Rappresentazione dei caratteri: normale, in negativo e lampeggiante.

GRAFICI
- Bassa risoluzione: 192Ø punti (4Ø x 48) oppure 16ØØ punti (4Ø x 4Ø con quattro linee di testo).
- 15 colori. Istruzioni Basic per la scelta dei colori.
- Alta risoluzione: 53.76Ø punti (28Ø x 192) oppure 44.8ØØ punti (28Ø x 16Ø con quattro linee di testo).
- 6 colori: bianco, nero, blù, verde, viola e arancio.

LA TASTIERA

La tastiera del LEMON II, ha la disposizione dei tasti secondo il modo QWERTY internazionale e comprende 64 tasti, compresi quelli di controllo cursore (solamente per gli spostamenti verso destra e verso sinistra), i tasti ESC, CTRL, SHIFT, RESET e un pratico tastierino numerico.

L'effetto del «REPEAT» è automatico su tutti i tasti.

COMANDI DI EDITING DELLO SCHERMO E DEL PROGRAMMA

Alimentando il vostro LEMON II, dopo alcuni secondi apparirà sul video la scritta «LEMON II» e una parentesi quadra chiusa «]» seguita dal cursore lampeggiante, corrispondente alla situazione di «pronto», cioè indica che il computer è in grado di ricevere delle istruzioni o dei comandi da eseguire.

Battendo a caso sulla tastiera mentre il cursore è attivo, tutto ciò che è stato battuto viene direttamente trasferito allo schermo. Battendo il tasto RETURN non si avrà nessun carattere stampato, ma si ottiene l'effetto del ritorno carrello della macchina per scrivere, posizionando il cursore all'inizio della linea successiva. A questo punto sullo schermo apparirà il messaggio ? SYNTAX ERROR e questo succede perchè una seconda funzione del tasto RETURN è quella di informare il sistema operativo BASIC che una linea è stata terminata e che si desidera che essa venga memorizzata. Perchè questo possa succedere occorre che nella linea scritta sullo schermo (e precisamente nella memoria video) siano presenti dei comandi esatti secondo la sintassi del linguaggio di programmazione e cioè del BASIC, in modo che questi vengano correttamenti interpretati ed eseguiti immediatamente.

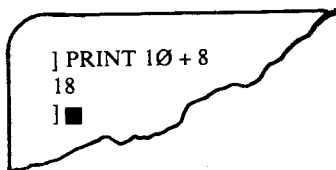
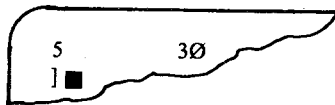
La risposta verrà visualizzata sullo schermo se la riga scritta non è stata preceduta da un numero.

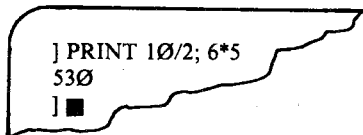
Questo modo di operare viene chiamato «modo DIRETTO» in contrapposizione a quello che viene chiamato «modo PROGRAMMATO».

Ogni riga (o linea) in «modo PROGRAMMATO» inizia con un «numero di linea», cioè un numero intero da Ø a 63999 compresi.

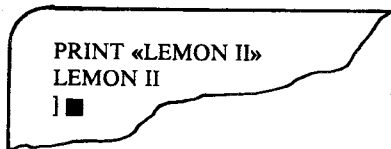
Una sequenza di comandi in modo programmato è detta «programma». Invece di eseguire immediatamente le istruzioni, il BASIC quando si batte il comando RUN e il tasto RETURN, esegue rapidamente dapprima l'istruzione col numero di linea più piccolo poi quella col numero di linea immediatamente superiore fino all'istruzione col numero di linea più elevato.

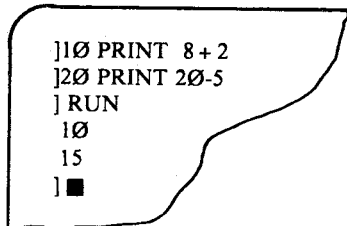
ESEMPIO DI ESECUZIONI DI COMANDI IN MODO DIRETTO

Provate a battere sulla tastiera quanto segue:
PRINT 1Ø + 8 e quindi premere il tasto RETURN.
LEMON II visualizzerà sullo schermo immediatamente il risultato come mostra l'esempio che segue:

NOTA: Ricordatevi di premere sempre il tasto RETURN al termine di ciascuna riga lunga o breve che sia.

Provate ora questo esercizio:
PRINT 1Ø/2, 6*5 RETURN (/ significa dividere, * significa moltiplicare).

Il BASIC esegue, in modo diretto, immediatamente le operazioni richieste.

Osservate la virgola «,» usata con il comando PRINT che ha la funzione di dividere la riga di 4Ø caratteri in tre colonne o campi di tabulazione di 16 spazi per i primi due e di 8 per il terzo.

Se al posto della «,» usate un «;» il valore successivo verrà stampato immediatamente dopo, in senso orizzontale.

Se si vogliono scrivere sullo schermo delle frasi, o commenti vari occorre inserirli fra apici (o virgolette), dopo il comando di stampa PRINT.

Ad esempio per scrivere sullo schermo la parola «LEMON II» occorre battere i seguenti comandi PRINT «LEMON II», come mostra l'esempio che segue:

ESEMPIO DI ESECUZIONI DI COMANDI IN MODO PROGRAMMATO

L'esecuzione «in modo programmato» si ottiene, come descritto precedentemente, facendo iniziare ogni linea con un numero intero da Ø a 63999, battendo il comando RUN per far partire il programma.

Provate questo esempio:

```
]1Ø PRINT  8 + 2
]2Ø PRINT 2Ø-5
] RUN
 1Ø
 15
]■
```

L'ordine in cui verranno scritte le righe non ha alcuna importanza in quanto il BASIC le mette in ordine per numero di linea progressivo.

Per controllare se ciò corrisponde a verità, provate a battere il comando LIST e vi apparirà il listato, cioè l'elenco delle linee, del programma completo come risiede in memoria.

```
] LIST
1Ø PRINT 8 + 2
2Ø PRINT 2Ø-5
]■
```

Nel caso voleste cancellare una linea intera dal programma, occorre battere il numero della linea seguito dal tasto RETURN.

Ad esempio:

Se invece di cancellare la linea numero 2Ø si voleva solamente sostituirla con una nuova occorre semplicemente scriverla però con il medesimo numero della linea stessa.

```
]2Ø PRINT 1ØØ-8Ø
]■
] LIST
1Ø PRINT 8 + 2
2Ø PRINT 1ØØ-2Ø
]■
```

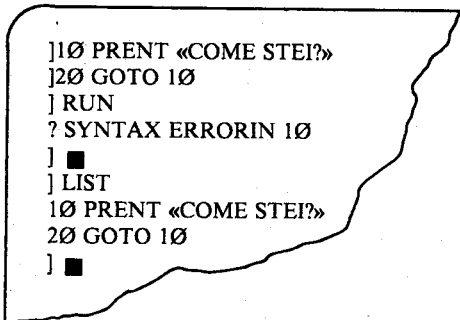
Quando si scrive per la prima volta un programma non è consigliabile numerare le righe consecutivamente, ma con un incremento di 1Ø per avere la possibilità di interporre nuove linee fra due esistenti se necessario.

Per cancellare l'intero programma battere il comando NEW.

Lo stesso comando verrà usato pure quando Voi volete scrivere un nuovo programma. Questo servirà per non mescolare vecchi e nuovi programmi.

USO DEL CURSORE

Per ottenere lo spostamento puro e semplice del cursore in ogni parte dello schermo, allo scopo di posizionarsi su una linea generica per eventuali modifiche, occorre abbinare il tasto ESC (escape) ai tasti A, B, C, D.

ESC A, sposta il cursore di uno spazio verso destra
ESC B, sposta il cursore di uno spazio verso sinistra
ESC C, sposta il cursore di uno spazio verso il basso
ESC D, sposta il cursore di uno spazio verso l'altro

Questi spostamenti non influenzano minimamente ciò che si trova sullo schermo e nella memoria.

Per un uso corretto premere prima il tasto ESC, quindi liberarlo e premere uno dei quattro tasti alfabetici.

Lo spostamento del cursore lo si può ottenere anche abbinando il tasto ESC ai tasti I, K, M, J; esso si sposterà verso l'alto, a destra, in basso e a sinistra in modo continuativo.

La correzione dei caratteri errati in una linea di programma si ottiene con i tasti → FRECCIA A DESTRA e ← FRECCIA A SINISTRA, tenendo presente che con la freccia a sinistra, si ottiene la cancellazione dalla memoria e non dallo schermo, con la freccia a destra si copia nella memoria il contenuto della linea che si trova sul video.

ESEMPIO DI CORREZIONE DI UNA LINEA DI PROGRAMMA

Per eseguire la correzione degli errori di battitura spostare il cursore sulla linea 1Ø battendo 3 volte il tasto ESC D e 1 volta ESC B per posizionare il cursore sul primo carattere. Premere ora 5 volte il tasto di → FRECCIA A DESTRA per posizionare il cursore sulla lettera O in «PRONT» e battere la I per correggere l'errore. Copiare il resto della linea sempre col tasto di → FRECCIA A DESTRA e posizionare il cursore sulla lettera E in «STEI» e battere la A per correggere. Continuare a copiare con → fino al termine della linea 1Ø e memorizzare la linea così corretta battendo RETURN.

Visualizzate il listato con il comando «LIST» per vedere il programma corretto.

```
] LIST
1Ø PRINT «COME STAI?»
2Ø GOTO 1Ø
] ■
```

Col comando «RUN» il programma incomincia a girare e lo si può arrestare premendo i tasti CTRL C. Lo schermo visualizza BREAK IN 1Ø (cioè arresto alla linea 1Ø, la linea che si sarebbe dovuta eseguire subito dopo). Vedi esempio:

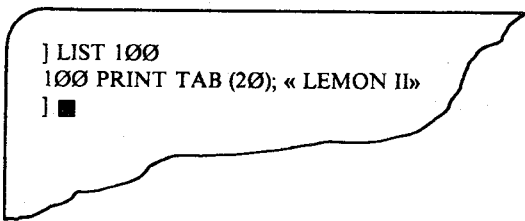
INSERIMENTO DI TESTO IN UNA LINEA ESISTENTE

Volendo inserire una parte di testo in una linea esistente procedere come segue:

1) Eseguire il LIST della linea da variare. Es:

Supponiamo di voler inserire dopo il PRINT, il comando TAB (2Ø).

2) Battere ESC B (per spostare il cursore verso sinistra) ed ESC D (che sposta il cursore verso l'alto) in modo da portare il cursore sul primo carattere della linea da variare, quindi usare il tasto di FRECCIA A DESTRA per copiare fino al primo segno di virgolette.

3) Battere ancora una volta ESC D per portare il cursore nella linea superiore vuota e scrivere il comando da inserire cioè TAB (2Ø). Battere ESC C per portare il cursore di una linea verso il basso, cioè si ritorna col cursore sulla linea 1ØØ.

4) Battere ESC B (e non il tasto con la freccia a sinistra che cancellerebbe i carat-teridell'inserimento appena battuti) fino al primo segno di virgolette. Lo schermo apparirà in questo modo:

5) Da questo punto usare il tasto con FRECCIA A DESTRA per copiare il resto della linea. Memorizzare la linea con l'inserimento battendo il tasto RETURN. Listando ora la linea 1ØØ dovrebbe apparire come segue:

ELEMENTI FONDAMENTALI DEL BASIC

NUMERI (COSTANTI)

In Basic le grandezze numeriche vengono indicate come numeri (o costanti).
Il formato dei numeri stampati dal BASIC segue le seguenti regole:

A I numeri sono rappresentati con una precisione di 9 cifre significative. Quando un numero viene stampato, vengono visualizzate soltanto 9 cifre. Ogni numero può avere anche un esponente, una potenza della base 1Ø.

B I numeri, in virgola mobile, devono essere compresi nel campo da -1*1Ø ^38 fino a 1*1Ø ^38 per non incorrere in errore, i valori interi devono comunque essere nel campo da -32767 a 32767.

C Se il numero è negativo è stampato il segno meno (-).

D Se il valore assoluto del numero è un numero intero che va da Ø a 999999999, esso viene stampato come numero intero.

E Se il valore assoluto del numero è superiore o uguale a .Ø1 e inferiore o uguale a 999999999.2, esso è stampato nella notazione in virgola fissa, senza esponente.

F Se il numero non cade nella regola «D» o «E», allora viene usata la notazione scientifica.

La notazione scientifica è formata come segue:
SX.XXXXXXXXESNN (ciascuna X rappresenta un numero intero da Ø a 9)
La S iniziale rappresenta il segno del numero, uno spazio se il numero è positivo e un segno meno (-) se il numero è negativo.
Un numero diverso da Ø è stampato prima del numero decimale.
Esso è seguito dal punto decimale e quindi dagli altri otto numeri della mantissa.
Quindi è stampata una (E) come esponente, seguita dal segno (S) dell'esponente; infine due numeri (NN) dell'esponente stesso. Lo zero (Ø) prima del punto decimale e quelli di coda, non significativi, non vengono mai stampati.
Se c'è soltanto un numero da stampare, dopo che tutti gli zeri di coda sono stati soppressi, viene soppresso anche il punto decimale.
L'esponente, composto sempre da due numeri con segno (+) se positivo e meno (-) se negativo; gli zeri nell'esponente non sono mai soppressi.
Il valore di un numero espresso in notazione scientifica è il numero alla sinistra di (E) moltiplicato per 1Ø elevato alla potenza indicata dal numero alla destra di (E).
I seguenti sono esempi di vari numeri e la relativa visualizzazione sullo schermo.

NUMERO	VISUALIZZAZIONE
+ 1	1
-1	-1
9999	9999
-99.99	-99.99

1*1Ø ^ 2Ø	1E + 2Ø
-12.34567897*1Ø ^ 1Ø	-1.2345679E + 11
1ØØØØØØ	1E + Ø6
999999999	999999999
Ø.1	.1
Ø.Ø1	.Ø1
Ø.ØØØ123	1.23E-Ø4

Una costante numerica usata in un programma, oppure un numero battuto sulla tastiera può avere un massimo di 38 cifre, ma solamente le prime dieci cifre sono significative e la decima è arrotondata.

Esempio:

VARIABILI

Variabile è un nome. che rappresenta un numero e una stringa. Ogni **variabile numerica** è costituita da una lettera alfabetica, o da una lettera seguita da un numero intero da Ø a 9 oppure da due lettere solamente.

ESEMPIO DI VARIABILI ACCETTATE DAL BASIC

A , B , C ,............. Z
AØ, A1, A2,.............A9
BØ, B1, B2,.............B9
....
....
ZØ, Z1, Z2,.............Z9
AA, AB, AC,.............AZ
BA, BB, BC,.............BZ
....
....
ZA, ZB, ZC,.............ZZ

Le **variabili a stringa** sono sequenze di caratteri racchiuse fra apici (« ») e si usano per rappresentare informazioni non numeriche: nomi, indirizzi, ecc..

Vengono usate anche come etichetta (label) a dati in uscita e per stampare delle risposte sullo schermo.

Le variabili a stringa vanno scritte come le variabili numeriche, seguite dal simbolo del dollaro ($)

Es:

A$, B$,.............Z$
AØ$, A1$,.............A9$
....

....
ZØ$, Z1$,.............Z9$
AA$, AB$,.............AZ$
....
ecc. ecc. ....

La lunghezza massima della stringa deve essere di 255 caratteri.
Provate ad eseguire, in MODO DIRETTO, la seguente assegnazione di una stringa:

Questa assegnazione, eseguita in MODO PROGRAMMATO, dà la possibilità di scrivere sullo schermo tutte le volte che occorre la frase (NOME DITTA) richiamando la stampa della stringa A$.

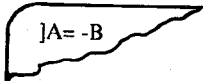
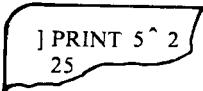
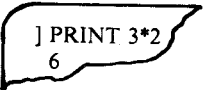
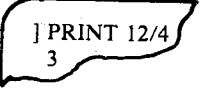
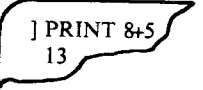
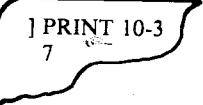
Un altro tipo di variabile che si può usare è la «variabile intera», che è rappresentata come la variabile numerica seguita dal carattere % (percento).

Es: A%, B% .... Z%
AØ%, A1% A9%
....
....
ZØ%, Z1% Z9%
AA%, BB% .... ZZ%

Le **variabili intere,** con numeri interi compresi nel campo da -32767 a +32767, vengono usate vantaggiosamente, nelle operazioni sulle matrici, per risparmiare spazio di memoria.

Per assegnare il valore ad una **variabile numerica** o ad una **variabile intera** si usa l'istruzione LET, la quale può anche essere omessa, facendo risparmiare tempo e memoria.

MATRICI

Le matrici sono delle variabili con indice che vengono usate spesso per semplificare il calcolo scientifico, perchè permettono di selezionare qualsiasi elemento in una tabella di numeri.

Le matrici possono avere un solo indice A (I), B (K) ecc. e vengono dette matrici

monodimensionali oppure due, tre, ecc. indici e vengono dette matrici a due, tre ecc. dimensioni.

Es: A (3,2), B (4,5) ecc..

Vedremo in seguito come riservare nella memoria del computer lo spazio necessario per le variabili e le matrici.

OPERATORI ALGEBRICI

SIMBOLI	ESEMPIO	DESCRIZIONE
=	]A= -B	**Negazione.** (N.B. A-B è una sottrazione mentre -A è una negazione).
^	] PRINT 5^2 25	**Elevazione ad esponente.** (eguale a 5*5)
*	] PRINT 3*2 6	**Moltiplicazione.**
/	] PRINT 12/4 3	**Divisione.**
+	] PRINT 8+5 13	**Addizione.**
-	] PRINT 10-3 7	**Sottrazione.**

Regole per elaborare le espressioni.

Moltiplicazioni e divisioni vengono eseguite prima delle addizioni e sottrazioni. Ad esempio 8+10/2= 13 e non 9.

Quando in una formula vengono incontrate operazioni con uguali precedenze, quelle più a sinistra sono eseguite prima.

Es: 6-4+3= 5 e non -1.

L'ordine in cui si desidera vengano eseguite certe operazioni al posto di altre, deve specificatamente essere espresso tramite l'uso di parentesi rotonde.

Ad esempio per sommare 5 a 9 e poi dividere per 2 scriveremo (5+9) /2= 7.

Se si fossero omesse le parentesi si sarebbe ottenuto un risultato sbagliato. Es: 5+9/2= 9,5.

ORDINE degli operatori gerarchicamente usato per elaborare le espressioni. Le operazioni sulla stessa riga hanno la stessa precedenza.

1) ( )	Le formule fra parentesi sono sempre elaborate per prime.
2)^	Elevazione a potenza.
3) -	Negazione. - X dove X può essere una formula.
4) * /	Moltiplicazione e divisione.
5) + -	Addizione e sottrazione.
6) operatori relazionali	= uguale < > diverso < minore di > maggiore di <= minore o uguale a >= maggiore o uguale a
7) NOT	«Negazione» logica a livello di bit
8) AND	«AND» logico a livello di bit
9) OR	«OR» logico a livello di bit

DESCRIZIONE DEI COMANDI BASIC

A fianco di ogni comando e delle varie istruzioni vengono indicati, tra parentesi i relativi modi di esecuzione:
D= comando eseguibile solamente in MODO DIRETTO
D & P= comando eseguibile in MODO DIRETTO e in MODO PROGRAMMATO
P= comando eseguibile solamente in MODO PROGRAMMATO

LIST (D & P)

LIST
Visualizza sullo schermo la lista dell'intero programma residente in memoria.

LIST 1ØØ-
Visualizza il programma della linea 1ØØ in poi.

LIST-1ØØ
Visualizza il programma fino alla linea 1ØØ.

LIST 1ØØ-2ØØ
Visualizza il programma dalla linea 1ØØ alla linea 2ØØ.

LIST X
Visualizza la linea del programma specificato da X.

REM (D & P)

1Ø REM= = calcolo area = =
Rem è l'abbreviazione di «remark», cioè «nota». Viene usata per inserire commenti o note in un programma e la linea viene ignorata dal Basic.
Si termina la linea col tasto RETURN.

DEL (D & P)

DEL 1ØØ, 2ØØ
Cancella dal programma in memoria le linee da 1ØØ a 2ØØ.
Per cancellare la sola linea 1ØØ usare DEL 1ØØ, 1ØØ oppure battere il numero 1ØØ e quindi il tasto RETURN.

VTAB (D & P)

1ØØ VTAB (Y)
Sposta il cursore sullo schermo solo nel senso verticale alla riga specificata da Y (da 1 a 24); 1 per la riga superiore e 24 per la riga inferiore.

HTAB (D & P)

11Ø HTAB (X)
Sposta il cursore sullo schermo a destra o a sinistra alla posizione specificata da X (da 1 a 255).

TAB (D & P)

12Ø PRINT TAB (2Ø) «*»
Usato sempre con il comando PRINT; la posizione iniziale di stampa viene specificata tra parentesi da un numero compreso tra Ø e 255.
PRINT TAB (Ø), pone il cursore nella posizione 256.
Nell'esempio, l'asterisco verrà stampato nella posizione 2Ø.

POS (D & P)

13Ø PRINT POS
Fornisce sullo schermo la posizione corrente, orizzontale del cursore rispetto al margine di sinistra, indicato col numero (Ø) mentre il margine destro è indicato dal numero (39) racchiuso fra parentesi rotonde.

HOME (D & P)

14Ø HOME
Cancella tutto lo schermo e posiziona il cursore nell'angolo superiore sinistro.

CLEAR (D & P)

15Ø CLEAR
Azzera tutte le variabili, le stringhe e le matrici ripristinando i puntatori e le cataste operative.

SPC (D & P)

16Ø PRINT SPC (2Ø)
Usato sempre con il comando PRINT, sposta il cursore di un numero di spazi bianchi, specificato tra parentesi, allontanandolo dalla parola precedentemente visualizzata.
Il numero degli spazi bianchi dello spostamento del cursore devono essere nel campo da Ø a 255. È possibile concatenare diversi SPC per creare grandi spazi.
Es: 17Ø PRINT SPC (2Ø) SPC (15Ø) SPC (119).

SPEED (D & P)

18Ø HOME: CLEAR: SPEED= 18Ø

Il numero con cui si vuol definire la velocità con la quale i caratteri devono essere trasmessi al video o ai dispositivi di input/output è separato dal comando «SPEED» dal segno di «uguale».

SPEED = Ø per la velocità più bassa
SPEED = 255 per la velocità più elevata.

FLASH (D & P)

185 E$= «LEMON II»
19Ø FLASH: PRINT E$

Definisce il lampeggio dei caratteri sul video presentandoli alternativamente bianchi su fondo nero e quindi neri su fondo bianco.
Nell'esempio la stringa E$ verrà visualizzata con effetto lampeggiante.

INVERSE (D & P)

2ØØ INVERSE: PRINT E$

Definisce l'inversione dei caratteri sullo schermo visualizzandoli neri su fondo bianco.

NORMAL (D & P)

21Ø NORMAL: PRINT E$

Ripristina al modo normale , lettere bianche sù fondo nero, la visualizzazione dei caratteri sullo schermo precedentemente disposta dai comandi FLASH e INVERSE.

FRE (D & P)

PRINT FRE (Ø)

Usato col comando PRINT visualizza sullo schermo il numero di bytes ancora disponibili per l'utente.
Nell'esempio è usato in modo DIRETTO.

CTRL (D)

CTRL X

Con questo comando viene ignorata la linea correntemente battuta senza cancellare una eventuale riga con lo stesso numero precedentemente battuta.
Al termine della linea da ignorare viene visualizzata una barretta invertita « \ » e il cursore si posiziona nella colonna Ø della linea seguente.

LOAD (D & P)

2ØØ LOAD

Carica (legge) un programma da nastro, nella memoria del LEMON II.
Riavvolgere il nastro e premere PLAY sul registratore, poi battere da tastiera LOAD.
Il comando non verifica se il registratore è presente o che sia nel modo corretto.

L'inizio e la **fine** della lettura è segnalata da un segnale acustico e dalla presentazione sullo schermo del carattere di pronto «]».
LOAD, quando inizia una nuova lettura dal nastro, cancella il programma corrente.
Per interrompere l'esecuzione usare il tasto RESET.

SAVE (D & P)

21Ø SAVE
Salva (memorizza) un programma su nastro se questo è stato correttamente posizionato.
Premere REC e PLAY contemporaneamente sul registratore, poi battere sulla tastiera il comando SAVE.
Anche questo comando non verifica se il registratore è presente o che siano stati premuti gli appropriati tasti del registratore.
L'inizio e la fine della registrazione è segnalata da un segnale acustico.
Per interrompere l'esecuzione usare il tasto RESET.

NEW (D & P)

22Ø NEW
Cancella il programma corrente comprese tutte le variabili.

RUN (D & P)

1) RUN
2) RUN 2ØØ
3) 23Ø RUN
1) Inizia l'esecuzione del programma in memoria iniziando dal numero di linea più basso.
2) Inizia l'esecuzione dalla linea specificata (nell'esempio dalla linea 2ØØ) in poi.
3) Comando «RUN» da programma.
Il RUN cancella tutte le variabili (fa un CLEAR) e riinizializza i DATA.

STOP (D & P)

3ØØ STOP
Ferma l'esecuzione del programma e attende una nuova istruzione.
Esso provoca la visualizzazione sul video del messaggio BREAK IN XX (dove XX è il numero dell'ultima linea eseguita del programma al momento del comando STOP.

END (D & P)

999 END
Comando di fine esecuzione di un programma senza visualizzare alcun messaggio di interruzione.
Può essere usato in qualunque punto del programma ed è facoltativo.

CTRL C (D)

CTRL C
Provoca l'arresto immediato del programma o di un listato.
Viene usato mantenendo premuto il tasto CTRL e battendo il tasto C.

RESET (D)

RESET
Interrompe immediatamente il programma in esecuzione.

CONT (D & P)

31Ø CONT
Provoca la ripresa dell'esecuzione alla successiva istruzione del programma dopo l'arresto provocato da STOP, END, CTRL C, RESET. Non è possibile continuare se si è commesso un errore.

CALL (D & P)

4ØØ CALL X
Provoca il salto e l'esecuzione di una subroutine in linguaggio macchina che inizia alla posizione di memoria specificato da X, dove X deve essere nel campo da -65535 a 65535.

PEEK (D & P)

45Ø PRINT PEEK (X)
Usato col comando PRINT visualizza il contenuto dell'indirizzo di memoria X.

46Ø X = PEEK (-16336)
Produce un «click» dell'altoparlante.

47Ø X = PEEK (-16352)
Abilita l'output sul registratore a cassetta per una volta e registra un «click».

48Ø X = PEEK (-16287)
Legge sul controllo del cursore # Ø, l'interruttore a pulsante. Se X > 127, quel pulsante viene premuto.

49Ø X = PEEK (-16286)
Legge, sul controllo del cursore # 1, l'interruttore a pulsante. Se X > 127, quel pulsante viene premuto.

5ØØ X = PEEK (-16285)
Pulsante per il controllo del sensore # 2.

51Ø X =PEEK (128) + PEEK (219) * 256
Se l'istruzione ONERRGOTO è stata eseguita, l'istruzione definisce X uguale al nu-

mero di linea in cui si è verificato l'errore.

52Ø IF PEEK (216) > 127 THENGOTO 1ØØØ
Se è stato definito vero il lit 7 nella posizione di memoria 222 (ERRFLG), ciò significa che è stata incontrata l'istruzione ONERRGOTO.

53Ø Y = PEEK (222)
Definisce un codice per la variabile Y descrivendo il tipo di errore che ha causato un salto ONERRGOTO:

CODICE	ERRORE	CODICE	ERRORE
Ø	NEXT senza FOR	12Ø	Matrice ridimensionata
16	Sintassi	133	Divisione per Ø
22	RETURN senza GOSUB	163	Variabile incoerente
42	Mancanza di DATA	176	Stringa troppo lunga
53	Quantità illecita	191	Formula troppo complessa
69	Superamento di capacità	224	Funzione non definita
77	Memoria non più disponibile	254	Risposta errata ad un INPUT
9Ø	Istruzione non definita	255	Tentativo di interruzione
1Ø7	Indice errato		

Segue una semplice subroutine in linguaggio macchina da usare in una routine di gestione dell'errore, con inizio dalla posizione 768.

54Ø POKE 768, 1Ø4: POKE 769, 168: POKE 77Ø, 1Ø4: POKE 771, 166: POKE 772, 223: POKE 773, 154: POKE 774, 72: POKE 775, 152: POKE 776, 72: POKE 777, 96.
Per l'utilizzo si userà il comando CALL 768.
Risolve problemi di ONERRGOTO con PRINT e messaggi «? OUT OF MEMORY ERROR.»

POKE (D & P)

6ØØ POKE X, Y
Il comando POKE memorizza il valore specificato nel secondo argomento (Y) nella posizione di memoria data dal primo argomento (X).
Il valore di Y deve essere = >Ø e < = 255.
Il valore di X deve essere = >Ø e < -65535 a 65535.
Il caricamento in una parte di memoria non utilizzata è innoquo ma molte posizioni di memoria contengono dati necessari al funzionamento del LEMON. Un POKE in queste posizioni di memoria può alterare il buon funzionamento del computer o del programma.

61Ø POKE - 16296,1
Definisce l'output di controllo # Ø, al TTL a collettore aperto alto (3,5 V.), connettore I/O per giochi spina 15. Questa è la condizione «off».

62Ø POKE - 16295,Ø
Definisce l'output di controllo # Ø al TTL basso (Ø,3 V.).
Questa è la condizione «on» a 1,6 milliampere.

63Ø POKE - 16294,1
Definisce l'output di controllo # 1 a TTL alto (3,5 V.) connettore I/O per giochi, spina 14.

64Ø POKE - 16293,Ø
Definisce l'output di controllo # 1 a TTL basso (Ø,3 V.).

65Ø POKE - 16292,1
Definisce l'output di controllo # Ø a TTL alto (3,5 V.), connettore I/O per giochi, spina 13.

66Ø POKE - 16291,Ø
Definisce l'output di controllo # 2 a TTL basso (Ø,3 V.).

67Ø POKE - 1629Ø,1
Definisce l'output di controllo # 3 a TTL alto (3,5 V.), connettore per giochi, spina 12.

68Ø POKE - 16289,Ø
Definisce l'output di controllo a TTL basso (Ø,3 V.).

69Ø POKE - 216,Ø
Cancella ERRFLG riattivando i messaggi di errore.

TRACE (D & P)

7ØØ TRACE
Visualizza sullo schermo il numero di ciascuna linea del programma man mano che viene eseguita.
Questo comando non è escluso da RUN, NEW, DEL, CLEAR o RESET.

NOTRACE (D & P)

71Ø NOTRACE
Annulla il comando TRACE.

HIMEM: (D & P)

72Ø HIMEM: X
Stabilisce l'indirizzo della posizione di memoria più alta disponibile per un programma, specificato da X, dove X è compreso nel campo da -65535 a 65535.
All'accensione, LEMON II definisce automaticamente HIMEM: sull'indirizzo di memoria

più elevato disponibile, corrispondente a 49152.
Nelle posizioni di memoria 116 e 115 decimali viene caricato il valore corrente di HIMEM: per visualizzare tale valore eseguire in modo diretto la seguente istruzione:
PRINT PEEK (116)*256+PEEK (115) RETURN, HIMEM: non viene influenzato dai comandi RUN, NEW, CLEAR, DEL, RESET e aggiungendo o cambiando una linea del programma.

LOMEM: (D & P)

73Ø LOMEM: X
Stabilisce l'indirizzo della posizione di memoria più bassa disponibile per un programma, specificato da X , dove X è compreso nel campo da -65535 a 65535.
ATTENZIONE al limite inferiore assoluto che deve essere posto circa a 2Ø51.
Il comando LOMEM: è ripristinato aggiungendo o cambiando una linea del programma e dai comandi NEW e DEL.
Nelle posizioni di memoria 1Ø6 e 1Ø5 decimali viene caricato il valore corrente di LOMEM:; per visualizzare tale valore eseguire in modo diretto la seguente istruzione:
PRINT PEEK (1Ø6)*256+PEEK (1Ø5) RETURN.

WAIT (D & P)

74Ø WAIT 1ØØØ, 2
Questo comando ritarda l'esecuzione della prossima istruzione del programma fino a che il bit 1 della locazione di memoria 1ØØØ decimale, diventa 1.
Il WAIT è molto più veloce dell'equivalente istruzione «IF» impiegando contemporaneamente minor numero di bytes per memorizzare il programma.

75Ø WAIT 1ØØØ, 255, Ø
Ritarda l'esecuzione della prossima istruzione del programma fino a che uno degli otto bits alla locazione 1ØØØ è alto, cioè 1.

76Ø WAIT 1ØØØ, 255, 255
Ritarda l'esecuzione della prossima istruzione del programma fino a che uno degli otto bits alla locazione 1ØØØ è basso, cioè Ø.

77Ø WAIT 1ØØØ, 1, 1
Come sopra, fino a che almeno il bit più a destra nella locazione 1ØØØ è basso cioè Ø, indipendentemente dalle condizioni degli altri bits.

78Ø WAIT 1ØØØ, 3, 2
Come sopra, fino a che il bit più a destra nella locazione 1ØØØ è alto cioè 1 o il successivo bit più a destra è basso cioè Ø o che esistono ambedue le condizioni.
Il valore del secondo e del terzo argomento devono essere nel campo da Ø a 255.

USR (D & P)

79Ø USR (X)
La funzione USR (X) trasferisce ad una subroutine in linguaggio macchina il valore dell'argomento X. Il valore dell'argomento viene valutato e caricato nell'accumulatore in virgola mobile, locazioni da $9D fino a $A3 e viene eseguito un JSR alla locazione $ØA.
Le locazioni da $ØA a $ØC devono contenere un JMP alla locazione di inizio della subroutine in linguaggio macchina.
Usare un RTS per ritornare in BASIC.

INPUT (P)

8ØØ INPUT A, B, C
Il comando INPUT, eseguibile solamente in modo programmato, visualizza sullo schermo un punto di domanda (?) e rimane in attesa di dati da tastiera.
Nell'esempio alla linea 8ØØ l'input attende l'ingresso da tastiera di tre dati numerici separati da virgole.

81Ø INPUT A$
Come sopra ma in attesa di un dato stringa.

82Ø INPUT «NUMERO»; A
Visualizza sullo schermo la stringa tra apice (NUMERO) e attende un dato numerico.

83Ø INPUT «NOME»;N$
Visualizza sullo schermo la stringa tra apici (NOME) e attende un dato a stringa.
Se la risposta ad un INPUT numerico non è un numero reale, un intero, una virgola o due punti, comparirà il messaggio:

? REENTER

facendo rieseguire l'INPUT.
La stessa cosa succederà se al posto della risposta all'INPUT, sia stato premuto il tasto RETURN.
Battendo il tasto RETURN come risposta ad un INPUT che prevedeva un'espressione a stringa, la risposta sarà interpretata come una stringa nulla e il programma continuerà la sua esecuzione.
Se viene battuto il tasto RETURN prima che siano state date tutte le risposte alle variabili, vengono visualizzati due punti di domanda (??) per indicare che è prevista una ulteriore risposta.
Se si sono battuti più dati di quelli richiesti, viene visualizzato il messaggio

? EXTRA IGNORED

e il programma continuerà la sua esecuzione.

GET (P)

84Ø GET A
Attende un singolo carattere da tastiera senza visualizzarlo sullo schermo e NON occorre premere il tasto RETURN.
Attenzione a battere solamente caratteri alfanumerici leciti, escludendo segni particolari, il punto, la virgola, i due punti ecc. altrimenti si cade in errore.

85Ø GET A$
Attende un singolo carattere a stringa da tastiera.
Questa istruzione dovrà essere usata anche per variabili aritmetiche convertendo poi la stringa in numero con la funzione VAL. Es:

```
1Ø GET A$
2Ø A = VAL (A$)
3Ø IF A = Ø GOTO 1Ø
```

PRINT (D & P)

PRINT 3 * 2
Questo è un comando di output verso lo schermo che permette la visualizzazione delle espressioni e i risultati della elaborazione dei dati manipolati.
L'esempio in **modo diretto** visualizza sullo schermo il risultato della moltiplicazione cioè 6.

86Ø PRINT
Il comando senza alcun argomento provoca una interlineatura sullo schermo.

87Ø PRINT A, B, C
Visualizza sullo schermo il valore delle variabili A, B, C rispettivamente nei campi di tabulazione stabilite dalla «virgola».
1° campo di tabulazione è di 16 posizioni (1 ÷ 16).
2° campo di tabulazione è di 16 posizioni (17 ÷ 32) ed è disponibile se nulla è stampato nella posizione 16.
3° campo di tabulazione è di 8 posizioni (33 ÷ 4Ø) ed è disponibile se nulla è stampato nelle posizioni da 24 a 32.

88Ø PRINT «NUMERO =»; A
Visualizza sullo schermo la stringa fra apici e di seguito il valore della variabile A.
La visualizzazione dell'esempio sarà:

```
NUMERO = 1Ø
(se A = 1Ø).
```

Se in una istruzione PRINT, due o più variabili numeriche o a stringa sono separate dal punto e virgola (;), i relativi valori vengono visualizzati concatenati, cioè uno di seguito all'altro senza nessuno spazio fra essi.
Es:

Il comando PRINT può essere abbreviato con il punto di domanda (?).

LET (D & P)

9ØØ LET A = 2Ø
LET è la cosiddetta «istruzione di assegnamento»: è usata per assegnare certi valori alle variabili.

91Ø B = 123
Per risparmiare tempo e memoria è possibile omettere LET scrivendo semplicemente B = 123.

92Ø A$ = «NOME»
L'assegnazione è permessa anche con variabili stringhe purchè racchiuse fra apici.
Es:

DATA (P)

94Ø DATA 1, -6, 1Ø
L'istruzione DATA, seguita da un elenco di dati numerici reali, interi, stringhe e costanti, viene usata vantaggiosamente in un programma quando questi dati non sono soggetti a cambiamenti, per cui possono essere usati continuamente nel corso del programma.
Le istruzioni DATA vengono poste in qualsiasi parte del programma e usate (lette) dall'istruzione READ.
I dati dovranno essere scritti nella linea coi DATA, nello stesso ordine che verrà usato dal programma.
Segue un breve listato in cui si può vedere l'uso delle istruzioni DATA, READ e RESTORE.
Nelle linee dei DATA possono essere inserite anche delle stringhe.
Se volete che la stringa contenga spazi bianchi iniziali, virgole, due punti, deve essere racchiusa tra due apici.
Es:

950 DATA « LEMON», GENNAIO, FEBBRAIO, « NUMERO 20»

```
90   HOME: CLEAR
100  INPUT «INDOVINA UN NUMERO»; N
110  READD
120  IFD=99999THEN190
130  IFD< >NTHEN110
140  PRINT«HAI INDOVINATO»
150  PRINT«VUOI CONTINUARE (S/N) ?»
160  GETA$ : IFA$< >«S»ANDA$< >«N»THEN160
170  IFA$=«S»THEN220
180  END
190  PRINT«SBAGLIATO ! ,PROVA ANCORA»
200  PRINT«PREMI UN TASTO»
210  GETA$ : IFA$=»»THEN210
220  RESTORE
230  GOTO90
240  DATA2,837,0,-13,90, 150,6.28
250  DATA69,3,10,44,99999
READY
```

READ (D & P)

960 READ V, W%, A$

Legge i valori nelle linee DATA assegnandoli alle variabili nell'istruzione READ.
Il primo dato letto sarà il primo dato nella prima linea dei DATA; il secondo letto sarà il secondo dato e così via.
Quando sono stati letti tutti i dati della prima linea dei DATA, i dati successivi verranno letti dalla seconda linea dei DATA e così via fino al termine delle linee dei DATA.
Se si tentasse di leggere (READ) più dati di quelli contenuti nelle linee dei DATA si cadrà in errore e apparirà il messaggio:

? OUT OF DATA ERROR IN (segue numero della linea del programma).

RESTORE (D & P)

970 RESTORE

Permette la rilettura, delle linee dei DATA.
Dopo l'istruzione RESTORE il primo dato letto sarà il primo dato della prima linea dei DATA. Il secondo dato letto sarà il secondo dato e così via come una normale operazione di lettura (READ).

: **I due punti** vengono usati per **separare** più istruzioni che si trovano sulla stessa linea di programma.

+ **Il segno «più»** è usato per il **concatenamento** di stringhe. La stringa risultante deve essere minore di 256 altrimenti si cadrà in errore.
  Es: 975 A$ = B$+C$

% **Il segno «percento»** è **l'identificatore** delle variabili intere di ampiezza compresa fra -32767 e 32767 (vedi variabili).

IN#... (D & P)

98Ø IN#X
Seleziona l'input dalla slot n° X dove X va da 1 a 7 compresi.
Specifica quale periferica fornirà l'input per le successive istruzioni INPUT.
IN#Ø predispone che l'input successivo avvenga dalla tastiera anzichè dagli slots.
Se viene selezionata una slot senza alcuna periferica, il sistema si arresta.
Per ripristinare, usare RESET.

PR# (D & P)

99Ø PR#X
Indirizza l'output alla slot X, dove X va da 1 a 7 compresi.
PR#Ø ripristina l'output allo schermo.
Se nella slot selezionata non è presente nessuna periferica il sistema si arresta.
Per ripristinare, usare RESET.

DEF (P)
FN (D & P)

1ØØØ DEF FN Z (A) = 2*A+A
Consente di definire, da parte dell'utente, delle funzioni che possono essere utilizzate successivamente nel programma.
Nell'esempio è definita la funzione FN Z (A) e verrà usata nel programma nella forma FN Z (15) oppure FN Z (5*3-1) ecc.
FN Z (15) fa si che 15 sostituisca A nell'espressione della funzione definita.
Nell'esempio della linea 1ØØØ, la funzione corrisponderà a 2*15+15 cioè 45.
Non è possibile definire funzioni a stringa.

GOTO (D & P)

GOTO XX
Fa si che il programma salti alla linea specificata dell'argomento XX, dove XX deve essere una linea presente nel programma, e l'esecuzione del programma prosegue da quella linea in poi.
Se la linea non esiste appare il messaggio:

? UNDEF'D STATEMENT ERROR IN (seguito dal numero di linea dove si è verificato l'errore).

GOSUB (D & P)

1Ø2Ø GOSUB XX
Fa si che il programma salti alla linea specificata dall'argomento XX, dove XX deve essere presente nel programma. Quando viene incontrata l'istruzione RETURN, il programma ritorna all'istruzione immediatamente successiva all'ultimo GOSUB eseguito.
Se la linea indicata nell'argomento non esiste viene presentato il messaggio:

? UNDEF'D STATEMENT ERROR IN (segue il numero di linea che contiene l'istruzione GOSUB).

Sono AMMESSI non più di 25 livelli nidificati di GOSUB, altrimenti viene presentato il messaggio:

? OUT OF MEMORY ERROR

RETURN (D & P)

1Ø3Ø RETURN
Questa è l'istruzione che fa ritornare il proseguimento del programma all'istruzione immediatamente successiva all'ultimo GOSUB eseguito.
La coppia di istruzioni GOSUB .... RETURN viene vantaggiosamente utilizzata, quando una parte del programma (subroutine) deve essere eseguita più volte, per evitare di ripetere le stesse istruzioni in diverse parti del programma.
Esempio di programma con l'uso delle istruzioni GOSUB.....RETURN:

```
1ØØ HOME
11Ø PRINT«DAMMI IL PRIMO NUMERO»;
12Ø GOSUB19Ø
13Ø A=N
14Ø PRINT«DAMMI IL SECONDO NUMERO»;
15Ø GOSUB19Ø
16Ø PRINT«LA SOMMA DEI DUE NUMERI È = »;A+N
17Ø STOP
18Ø REM = = = SUBROUTINE = = =
19Ø INPUTN
2ØØ IFN=INT(N)THEN23Ø
21Ø PRINT«DAMMI SOLO NUMERI INTERI»
22Ø GOTO19Ø
23Ø RETURN
```

POP (D & P)

1Ø4Ø POP
Questa istruzione ha l'effetto di un RETURN ma senza provocare il salto
Incontrando il successivo RETURN anzichè ritornare all'istruzione immediatamente successiva all'ultimo GOSUB eseguito, provoca il proseguimento del programma all'istruzione immediatamente successiva al SECONDO GOSUB eseguito (cioè al penultimo GO-

SUB eseguito).

IF...THEN (D & P)

1Ø5Ø IF A> 1Ø THEN 1ØØ
L'istruzione IF è usata per saltare ad una data istruzione solo se si è verificata una certa relazione.
Nell'esempio della linea 1Ø5Ø, il programma salterà alla linea 1ØØ solo se il valore della variabile A è maggiore di 1Ø. In caso contrario, continuerà l'esecuzione alla linea successiva.
La stessa linea può essere scritta, ottenendo lo stesso risultato, come segue:

- 1Ø5Ø IF A> 1Ø THEN GOTO 1ØØ
- 1Ø5Ø IF A> 1Ø GOTO 1ØØ

Ecco un breve esempio con l'istruzione IF...THEN.

```
1ØØ  HOME
11Ø  INPUT«BATTI DUE NUMERI SEPARATI DA UNA VIRGOLA»;N,M
12Ø  IF A<=B THEN 15Ø
13Ø  PRINT«N È MAGGIORE»
14Ø  GOTO 1ØØ
15Ø  IF N<M THEN GOTO 18Ø
16Ø  PRINT«NUMERI UGUALI»
17Ø  GOTO 1ØØ
18Ø  PRINT«M È MAGGIORE»
19Ø  GOTO 1ØØ
```

Nell'istruzione IF....THEN è possibile usare indistintamente tutti gli operatori relazionali.

FOR...NEXT (D & P)

1Ø6Ø FOR A = 1TO5ØØ:NEXT
FOR...NEXT consente di eseguire calcoli ripetitivi in successione delle linee di programma comprese fra le istruzioni FOR e NEXT, dove NEXT rappresenta la fine del blocco e deve sempre essere presente.
Nell'istruzione FOR vengono specificati una variabile (B), un valore iniziale (1), e un valore finale (5).
Le linee del programma comprese tra le istruzioni FOR e NEXT vengono eseguite più volte, iniziando con il valore iniziale assegnato a B e quindi rincrementando B di 1 fino a quando non si è raggiunto il valore finale; poi il programma prosegue regolarmente.
Il valore iniziale e finale può essere espresso anche con variabili ed espressioni (es: 2ØØ FOR A=MTO 1Ø*C). Naturalmente il valore finale deve essere uguale o maggiore (=>) del valore iniziale.
Provate ad eseguire il semplice esempio che segue per familiarizzare con queste istruzioni che formano i cosidetti «Loops».

Nell'istruzione FOR ... NEXT ora descritta la variabile di conteggio (B) veniva incrementata di 1 ad ogni ciclo o «interazione».
A volte ciò può non bastare, tornando utili diversi incrementi o decrementi, per cui si usa il comando STEP.

Es: 1ØØ FOR I=1TO5Ø...NEXT (Si ha un incremento di 1 unità ad ogni ciclo).
11Ø FOR I=1TO5Ø STEP 1Ø...NEXT (Si ha un incremento di 1Ø unità ad ogni ciclo).
12Ø FOR I=1ØTO Ø STEP -2...NEXT (Si ha un decremento di 2 unità ad ogni ciclo).

Si possono eseguire «Loops» FOR...NEXT indicati, uno dentro l'altro, facendo però attenzione che essi non si incrocino e non più di 1Ø, altrimenti si cade in errore.
Un solo NEXT può essere usato per chiudere più cicli FOR.

Es: 15Ø NEXT A, D, Z

ON...GOTO... (P)

1ØØ ON X GOTO 5ØØ, 6ØØ, 7ØØ, 8ØØ
L'istruzione esegue un GOTO alla linea indicata all'X- esimo numero dopo GOTO, cioè:

Esempio della linea 1ØØ:
- Se X = 1 salta alla linea 5ØØ
- se X = 2 salta alla linea 6ØØ
- se X = 3 salta alla linea 7ØØ
- se X = 4 salta alla linea 8ØØ

Se X = o maggiore (>) del numero delle linee alternative elencate, viene eseguita la linea successiva a queste.
L'espressione X deve essere nel campo da Ø a 255 altrimenti si cade in errore.

ON...GOSUB (P)

2ØØ ON X GOSUB 1ØØ, 2ØØ, 3ØØ, 4ØØ

Questa istruzione funziona in modo analogo a ON...GOTO, ma viene eseguito un GOSUB anzichè un GOTO.

ONERR GOTO (P)

3ØØ ONERR GOTO X

Può essere usato, quando in un programma si verifica un errore, per evitare la comparsa del messaggio di errore che interrompe l'esecuzione.

Il comando ONERR GOTO provoca un salto incondizionato alla linea di programma indicata da X. POKE 216, Ø ripristina. il sistema in modo da riavere la comparsa dei normali messaggi di errore.

Eseguendo PRINT PEEK (222) si ha la possibilità di vedere in quale errore è caduto il programma (vedi comando PEEK (222)).

RESUME (P)

Consente la continuazione dell'esecuzione del programma dall'inizio dell'istruzione in cui è caduto in errore.

Questo comando viene usato al termine della routine di gestione dell'errore.

DIM (D & P)

15Ø DIM A (4), B (1Ø), M (2Ø, 2Ø)

Comando di assegnazione, nella memoria, degli spazi per le matrici.

Tutti gli elementi delle matrici vengono inizializzati a Ø.

Il BASIC riserva automaticamente nella memoria, anche se non specificato con il comando DIM, uno spazio per le variabili che non abbiano più di 11 elementi.

L'istruzione DIM può comparire in qualsiasi linea del programma, prima dell'istruzione END, ma è preferibile porla all'inizio del programma.

CHR$ (D & P)

16Ø PRINT CHR$ (X)

Visualizza sullo schermo il carattere ASCII corrispondente al valore specificato da X, dove X è compreso tra Ø e 255.

ASC (D & P)

17Ø PRINT ASC (X$)

Visualizza sullo schermo il valore numerico ASCII del primo carattere della stringa X$.

VAL (D & P)

18Ø PRINT VAL (X$)

Visualizza sullo schermo il valore dell'espressione stringa X$ convertita in un numero. Il primo carattere della stringa deve essere di tipo numerico, altrimenti viene visualizzato uno «Ø».

LEN (D & P)

19Ø PRINT LEN (X$)
Visualizza sullo schermo la lunghezza, in caratteri, della stringa X$, con numeri compresi tra Ø e 255.
I caratteri non stampati e gli spazi bianchi vengono conteggiati nella lunghezza totale.

STR$ (D & P)

2ØØ PRINT STR$ (X)
Con questa funzione si ottiene la conversione dell'espressione numerica X in una stringa.
Es: 21Ø PRINT STR$ [25,4), dà la stringa «25,4».

LEFT$ (D & P)

22Ø PRINT LEFT$ (X$, I)
Visualizza gli I caratteri più a sinistra della stringa X$.
Se I < 1 o > 255 si cade in errore e verrà visualizzato il messaggio:

? ILLEGAL QUANTITY ERROR

Esempio:

RIGHT $ (D & P)

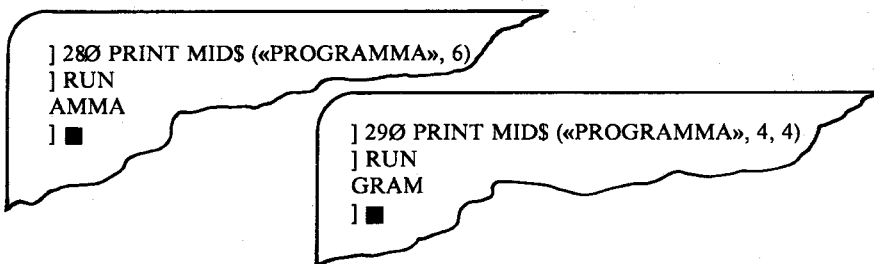
24Ø PRINT RIGHT $ (X$, I)
Visualizza gli I caratteri più a destra della stringa X$.
Se I < 1 o > 255 si cade in errore e verrà visualizzato il messaggio:

? ILLEGAL QUANTITY ERROR

Esempio:

MID$ (D & P)

26Ø PRINT MID$ (X$, I)
27Ø PRINT MID$ (X$, I,J)

Se eseguito con 1 argomento, visualizza sullo schermo i caratteri della stringa X$ iniziando dal carattere nella posizione specificato da I fino all'ultimo carattere, procedendo verso destra.
Se eseguito con 2 argomenti, visualizza sullo schermo i caratteri della stringa X$ iniziando dal carattere nella posizione specificato da I fino al carattere nella posizione specificato da J.
Esempio:

Se I < 1 o > 255 si cadrà in errore visualizzando il messaggio:

? ILLEGAL QUANTITY ERROR

STORE (D & P)

Il comando permette la registrazione su nastro di una matrice reale o intera.
Quando il comando STORE viene eseguito non viene dato nessun segnale o richiesta e occorre premere RECORD e PLAY nel registratore.
L'inizio e la fine della registrazione viene segnalata da segnali acustici.
Prima di registrare su nastro una matrice con STORE, occorre averla dimensionata con DIM.
Esempio:

```
3ØØ DIM M (4,18)
31Ø M = 5
32Ø STORE M
```

memorizza sul nastro i 95 (5*19) elementi della matrice M; da M (Ø,Ø) a M (4,18) senza cambiare il valore della variabile M.
Per poter memorizzare matrici di stringhe occorre prima convertirle in interi tramite la funzione ASC.

RECALL (D & P)

Il comando richiama un numero reale o una matrice di interi da nastro. È possibile richia-

mare con RECALL una matrice con un nome diverso da quello usato nella memorizzazione con STORE. Quando viene richiamata una matrice, occorre che sia stato eseguito nel programma il dimensionamento con DIM.
Quando il comando RECALL viene eseguito non viene dato nessun segnale o richiesta, occorre premere PLAY nel registratore.
L'inizio e la fine della lettura viene segnalata da un segnale acustico.
Esempio:

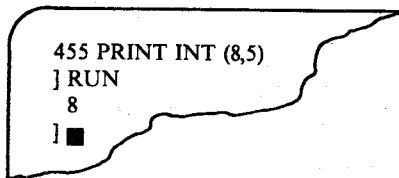
```
4ØØ DIM M (4,18)
41Ø M = 5
42Ø RECALL M
```

Legge da nastro i 95 (5*19) elementi della matrice M; da M (Ø,Ø) a M (4,18), senza cambiare il valore della variabile M.

FUNZIONI DI LIBRERIA

Le funzioni che seguono sono sempre disponibili, sia nel MODO DIRETTO che nel MODO PROGRAMMATO.

INT (D & P)

45Ø PRINT INT (X)
Visualizza sullo schermo il più grande numero intero uguale o minore di X.
Esempio:

ABS (D & P)

46Ø PRINT ABS (X)
Visualizza il valore assoluto dell'espressione X.
Dà -X se X è uguale o minore di Ø
dà X se X è maggiore di Ø

Es: PRINT ABS (-25.3) = 25.3

SGN (D & P)

47Ø PRINT SGN (X)
Visualizza -1 se l'argomento X è negativo, Ø se l'argomento X è Ø e 1 se l'argomento X è positivo.
Esempio:

SGN (3,14) = 1
SGN (Ø) = Ø
SGN (-1,25) = -1

EXP (D & P)

47Ø PRINT EXP (X)
Visualizza la costante «E» (2.718289) elevata all'esponente X con una approssimazione di sei cifre decimali.

SQR (D & P)

48Ø PRINT SQR (X)
Visualizza il risultato della radice quadrata dell'argomento X.
Se X è minore di Ø si cadrà in errore.

SIN (D & P)

49Ø PRINT SIN (X)
Visualizza il seno dell'espressione X in radianti.

COS (D & P)

5ØØ PRINT COS (X)
Visualizza il coseno dell'espressione X in radianti.

TAN (D & P)

51Ø PRINT TAN (X)
Visualizza la tangente dell'espressione X in radianti.

ATN (D & P)

52Ø PRINT ATN (X)
Visualizza l'arcotangente in radianti dell'espressione X.
Il risultato è compreso tra $-\pi/2$ e $\pi/2$.

LOG (D & P)

53Ø PRINT LOG (X)
Visualizza il logaritmo naturale (base E) dell'espressione X.
Per ottenere il logaritmo in base Y di X usare la formula LOG (X)/LOG (Y).

Esempio:

Logaritmo di 7 in base 1Ø = LOG (7)/LOG (1Ø).

RND (D & P)

54Ø PRINT RND (X)

Visualizza sullo schermo un numero reale casuale tra Ø e 1.
Se X è maggiore di Ø da inizio ad una nuova sequenza di numeri casuali ogni qualvolta viene usato.
Se X è minore di Ø, il comando RND genera lo stesso numero casuale ogni qualvolta viene usato con la stessa espressione X.
Se X è uguale a Ø, RND dà l'ultimo numero casuale precedente generato.
Generalmente l'espressione X è posta a 1.
L'esempio che segue mostra come eseguire la generazione di numeri casuali interi tra 6 e 1.

LE FUNZIONI CHE SEGUONO, ANCHE SE NON INCORPORATE NEL BASIC, POSSONO ESSERE CALCOLATE DERIVANDOLE DA QUELLE ESISTENTI E VOLENDO SI POSSONO FACILMENTE IMPLEMENTARE CON DEF FN.

SECANTE: SEC (X) = 1/COS (X)
COSECANTE: CSC (X) = 1/SIN (X)
COTANGENTE: COT (X) = 1/TAN (X)
SENO INVERSO: ARCSIN (X) = ATN (X/SQR (-X*X+1))
COSENO INVERSO: ARCCOS (X) = -ATN (X/SQR (-X*X+1)) +1.57Ø8
SECANTE INVERSA: ARCSEC (X) = ATN (SQR (X*X-1)) + (SGN (X)-1) * 1.57Ø8
COSECANTE INVERSA: ARCCSC (X) = ATN (1/SQR (X*X-1)) + (SGN (X)-1) * 1.57Ø8
COTANGENTE INVERSA: ARCCOT (X) = -ATN (X) + 1.57Ø8
SENO IPERBOLICO: SINH (X) = (EXP (X) -EXP (-X))/2
COSENO IPERBOLICO: COSH (X) = (EXP (X) + EXP (-X))/2
SECANTE IPERBOLICA: SECH (X) = 2/ (EXP (X) + EXP (-X))
COSECANTE IPERBOLICA: CSCH (X) = 2/(EXP (X) -EXP (-X))
TANGENTE IPERBOLICA: TANH (X) = -EXP (-X)/(EXP (X) + EXP (-X))*2+1
COTANGENTE IPERBOLICA: COTH (X) = EXP (-X)/(EXP (X) -EXP (-X))*2+1
SENO IPERBOLICO INVERSO: ARGSINH (X) = LOG (X+SQR (X*X+1))
COSENO IPERBOLICO INVERSO: ARGCOSH (X) = LOG (X+SQR (X*X-1))
TANGENTE IPERBOLICA INVERSA: ARGTANH (X) = LOG ((1+X)/(1-X))/2
SECANTE IPERBOLICA INVERSA: ARGSECH (X) = LOG (SQR (-X*X+1)+1)/X
COSECANTE IPERBOLICA INVERSA: ARGCSCH (X) = LOG (SGN (X)*SQR (X*X+1)+1)/X
COTANGENTE IPERBOLICA INVERSA: ARGCOTH (X) = LOG ((X+1)/(X-1))/2
A MOD B: MOD (A) = INT ((A/B-INT (A/B))*B+.Ø5)*SGN (A/B)

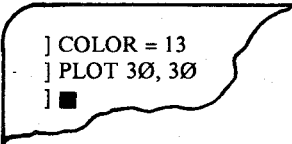
PDL (D & P)

25Ø PDL (X)

Questo è il comando per i trasduttori a manopola o sensori o PADDLE usati per pilotare il LEMON nei videogames.
Il valore specificato da X, che deve essere compreso da Ø a 3, specifica il numero del sensore scelto dandone il suo valore corrente da Ø a 255.
I sensori hanno una resistenza variabile da Ø a 15Ø Kohm.
Per ottenere delle accurate letture dei sensori con istruzioni PDL consecutive, occorre far eseguire al programma un breve loop di ritardo del tipo FORI = 1TO1Ø: NEXTI, altrimenti la lettura del secondo sensore può essere influenzata da quella del primo.

GR (D & P)

2ØØØ GR

Questo comando abilita il modo grafico a colori a BASSA RISOLUZIONE (4Ø X 4Ø), oscura lo schermo e riservando 4 linee per il testo sulla parte inferiore di esso, definisce in NERO il successivo colore da tracciare.
Facendo seguire a GR il comando POKE-163Ø2,Ø oppure il comando equivalente POKE 49234,Ø si ha l'esecuzione di grafici su schermo intero (4Ø X 48).
Per ripristinare l'esecuzione al modo grafico più testo far seguire al comando POKE a schermo intero, il comando GR.
Dopo ogni comando GR, il colore dello schermo si predispone a Ø cioè NERO.

COLOR = (D & P)

21ØØ COLOR = X

Il comando definisce nel modo grafico a colori a BASSA RISOLUZIONE, il colore con cui verrà tracciato il grafico è specificato da X, dove X va da Ø a 15 con i seguenti colori:
Ø = Nero, 1 = Magenta, 2 = Blu scuro, 3 = Porpora, 4 = Verde scuro, 5 = Grigio, 6 = Blu medio, 7 = Azzurro, 8 = Marrone, 9 = Arancio, 1Ø = Grigio chiaro, 11 = Rosa, 12 = Verde, 13 = Giallo, 14 = Celeste, 15 = Bianco.
Se si usa il proprio televisore a colori occorre eseguire, se questo non è perfettamente sintonizzato, un piccolo ritocco alla saturazione del colore e al contrasto; per fare questo eseguite il seguente test:

I due comandi tracciano un quadrato giallo al centro dello schermo e il colore deve corrispondere ad un giallo limone chiaro; contrariamente eseguire le necessarie regolazioni.
Il breve programma che segue è un ulteriore TEST DEL COLORE per una migliore messa a punto del vostro televisore.
Battere quanto segue e dopo il consueto controllo visivo, fatelo girare, col comando RUN.

```
] 1ØØ GR: HOME
  11Ø FOR I = ØTO31
  12Ø COLOR = I/2
  13Ø VLIN Ø,39 AT I
  14Ø NEXT I
  15Ø FOR I = ØTO 14 STEP 2: PRINT TAB (I*2+1); I; :NEXT I
  16Ø PRINT
  17Ø FOR I = 1TO15STEP2: PRINT TAB (I*2+1); I; :NEXT I
  18Ø PRINT: PRINT «TEST DEL COLORE»;
] ■
```

SCRN (D & P)

215Ø SCRN (X, Y)
Quando è usato nel modo grafico a bassa risoluzione, dà il codice del colore nel punto specificato da X, Y, dove X è compreso tra Ø e 39 e Y tra Ø e 47.

TEXT (D & P)

22ØØ TEXT
Ripristina al modo di testo normale a schermo intero (4Ø caratteri e 24 linee) dal modo grafico a bassa risoluzione oppure dal modo grafico ad alta risoluzione a schermo intero (28Ø X 192), al modo grafico ad alta risoluzione più 4 linee di testo (28Ø X 16Ø) e viceversa.

PLOT (D & P)

23ØØ PLOT X, Y
Traccia, nel modo grafico a bassa risoluzione, un punto nella posizione specificata in X, Y; dove X deve essere compreso tra Ø e 39, e Y compreso tra Ø e 47.
L'angolo superiore sinistro dello schermo è posto a Ø,Ø per tutti i modi grafici.

HLIN (D & P)

24ØØ HLIN X, Y AT Z
Nel modo grafico a bassa risoluzione disegna una linea orizzontale sullo schermo nella posizione specificata da X, Y e Z, dove X e Y devono essere comprese tra Ø e 39, Z tra Ø e 47.

VLIN (D & P)

25ØØ VLIN X, Y AT Z
Nel modo grafico a bassa risoluzione disegna una linea verticale sullo schermo nella posizione specificata da X, Y e Z, dove X e Y devono essere compresi tra Ø e 47, Z tra Ø e 39.

HGR (D & P)

26ØØ HGR
Questo è il comando che definisce il MODO GRAFICO AD ALTA RISOLUZIONE (28Ø X 16Ø), riservando sulla parte inferiore dello schermo 4 linee per il testo e corrisponde alla pagina 1 della memoria.
Il comando oscura lo schermo, lasciando la finestra di testo a schermo intero, visualizzando solamente le 4 linee inferiori e il cursore rimane sempre nella finestra di testo senza essere visibile, a meno che non lo si sposti su una delle 4 linee inferiori.
Per convertire lo schermo nel modo grafico ad alta risoluzione a schermo intero (28Ø X 192) occorre eseguire, DOPO HGR il comando POKE -163Ø2,Ø o l'equivalente POKE 49234,Ø.

HGR2 (D & P)

27ØØ HGR2
Definisce il modo grafico ad ALTA RISOLUZIONE a schermo intero (28Ø X 192) e corrisponde alla pagina 2 della memoria.
Il comando oscura lo schermo e la memoria dello schermo di testo non viene influenzata.
Il comando POKE -163Ø1,Ø viene usato per convertire QUALSIASI MODO GRAFICO A SCHERMO INTERO in un modo MISTO cioè GRAFICO + TESTO.

HPLOT (D & P)

28ØØ HPLOT X, Y
Traccia, nel modo grafico ad alta risoluzione, un punto nella posizione specificato in X, Y. Il colore del punto scelto è quello stabilito dall'istruzione HCOLOR più recente.

281Ø HPLOT TO X, Y, traccia una linea sullo schermo a partire dall'ultimo punto tracciato fino al punto determinato da X, Y. Il colore della linea è quello stabilito dall'istruzione HCOLOR più recente.

282Ø HPLOT X, Y TO X1, Y1, traccia una linea sullo schermo a partire dal punto X, Y fino al punto X1, Y1. Il colore della linea è quello stabilito dall'istruzione HCOLOR più recente.
Esiste la possibilità di estendere la linea, aggiungendo nella stessa istruzione altri comandi del tipo: TO X2, Y2 TO X3, Y3 TO X4, Y4 e così via fino al limite di 255 caratteri dell'istruzione.
Il valore dell'argomento deve essere compreso nel campo da Ø a 279 per X, e da Ø a 191 per Y.
Per un corretto uso di HPLOT, occorre precederlo da HGR o HGR2 per evitare di danneggiare una parte della memoria, del programma e le variabili.
Per tracciare una cornice rettangolare intorno ai 4 lati dello schermo ad alta risoluzione, usare la seguente istruzione:
283Ø HPLOT Ø,Ø TO 279,Ø TO 279,159 TO Ø,159 TO Ø,Ø.

HCOLOR (D & P)

284Ø HCOLOR = X

Il comando definisce nel modo grafico ad ALTA RISOLUZIONE, il colore con cui verrà tracciato il grafico e specificato da X, dove X va da Ø a 7 con i seguenti colori:

Ø = Nero 1, 1 = Verde, 2 = Blu, 3 = Bianco 1, 4 = Nero 2, 5 = Viola, 6 = Arancio, 7 = Bianco 2.

Il colore dipende dal tipo di televisore usato.

PEEK, POKE, CALL

FINESTRA DI TESTO

Utilizzando i quattro comandi POKE che seguono è possibile determinare la finestra di testo, in cui il testo stesso viene visualizzato e fatto ruotare sullo schermo. Ciò non cancella il resto dello schermo e non posiziona il cursore nella finestra di testo; per questi scopi si dovranno usare i comandi HOME oppure HTAB e VTAB.

PER NON DANNEGGIARE IL PROGRAMMA E IL BASIC OCCORRE OSSERVARE RIGOROSAMENTE I CONFINI DELLA FINESTRA DI TESTO: MAX 4Ø caratteri per 24 linee.

1ØØ POKE 32,S
Il comando determina il margine di sinistra dello schermo al valore specificato da S, dove S è compreso tra Ø e 39.
La posizione più a sinistra è determinata dallo Ø.

11Ø POKE 33, M
Il comando determina la giustezza, cioè il numero dei caratteri per linea da visualizzare, specificata da M, dove M è compresa tra 1 e 4Ø.
ATTENZIONE: NON DEFINIRE M uguale a Ø POICHÈ SI AVRÀ LA DISTRUZIONE DEL BASIC.

12Ø POKE 34, V
Il comando determina il margine superiore dello schermo al valore stabilito da V, dove V è compreso tra Ø e 23.
Ø = linea superiore; 23 = linea inferiore.
Non definire il margine superiore V al di sotto del margine inferiore I.

13Ø POKE 35, I
Il comando determina il margine inferiore dello schermo al valore specificato da I, dove I è compreso tra Ø e 24.
La linea inferiore dello schermo è 24.
Non definire il margine inferiore I al di sopra del margine superiore V.

14Ø CH = PEEK (36)
Questo comando legge la posizione corrente **orizzontale** del cursore e fissa la variabile CH uguale a tale posizione.
CH sarà compresa nel campo da Ø a 39 che è la posizione del cursore rispetto al margine di sinistra definito da POKE 32,S.

15Ø CV = PEEK (37)
Viene letta la posizione corrente **verticale** del cursore e fissa la variabile CV uguale a tale posizione.

CV sarà compresa nel campo da Ø a 23.

16Ø POKE 36, CH
Posiziona il cursore a CH + 1 posizioni a partire dal margine di sinistra, il quale è situato a POKE 36,Ø.
CH sarà < = (minore di o uguale a) alla larghezza definita da POKE 33,M e > = (maggiore di o uguale) a Ø.

17Ø POKE 37,CV
Posiziona il cursore alla posizione verticale specificata da CV.
CV = Ø definisce la riga superiore
CV =23 definisce la riga inferiore.

7ØØ POKE - 163Ø4,Ø
Visualizza sullo schermo un modo per grafici.

71Ø POKE - 163Ø3,Ø
Visualizza sullo schermo un modo testo.

72Ø POKE - 163Ø2,Ø
Visualizza sullo schermo tutta la pagina per grafici o tutta la pagina per testo.

73Ø POKE - 163Ø1,Ø
Unisce il modo testo con il modo grafico.

74Ø POKE - 163ØØ,Ø
Visualizza sullo schermo la prima pagina.

75Ø POKE - 16299,Ø
Visualizza sullo schermo la seconda pagina.

76Ø POKE - 16298,Ø
Visualizza sullo schermo il modo grafico in BASSA RISOLUZIONE.

77Ø POKE - 16297,Ø
Visualizza sullo schermo il modo grafico in ALTA RISOLUZIONE.

CALL X (D & P)

18Ø CALL - 936
L'istruzione dell'esempio di cui sopra, provoca la cancellazione dello schermo posizionando il cursore in alto a sinistra.
L'effetto provocato è lo stesso del comando ESC @ e del comando HOME.

19Ø CALL - 868
Provoca la cancellazione della riga corrente dal cursore fino al margine destro.
L'effetto è lo stesso di ESC E.

200 CALL - 912
Provoca lo spostamento verticale verso l'alto di una linea del testo.
La linea superiore di conseguenza scompare dallo schermo e la precedente seconda linea diventa la prima.

210 CALL - 922
Provoca l'esecuzione di un'interlinea.
Lo stesso risultato lo si ottiene anche col CTRL J.

220 CALL - 958
Provoca la cancellazione dallo schermo di tutti i caratteri a partire dalla posizione corrente del cursore fino al margine inferiore.
I caratteri situati alla sinistra e sulla stessa riga del cursore e le righe al di sopra di questo non vengono interessati.
Lo stesso risultato lo si ottiene anche con ESC F.

230 CALL - 1994
Nel modo grafico a BASSA RISOLUZIONE provoca l'oscuramento delle 40 linee superiori dello schermo mantenendo inalterato il contenuto della pagina di testo 2 e il modo grafico ad alta risoluzione.
Se usato nel modo TEXT sostituisce le prime 20 linee con segni@nero su bianco.

240 CALL - 1998
Nel modo grafico a schermo intero a BASSA RISOLUZIONE provoca l'oscuramento dello schermo mantenendo inalterato il contenuto dellapagina di testo 2 e il modo grafico ad alta risoluzione.
Se usato nel modo TEXT sostituisce l'intera pagina con segni @

250 CALL 62450
Nel modo grafico ad ALTA RISOLUZIONE provoca la cancellazione dallo schermo, dell'ultima pagina usata.

260 CALL 62454
Il comando ripristina lo schermo ad ALTA RISOLUZIONE tenendo conto dello schermo che è stato usato per ultimo ed usando l'ultimo colore tracciato.
Occorre far precedere questo comando da un tracciato.

270 X = PEEK (-16384)
Il comando legge la tastiera. Se X è maggiore di 127 significa che un tasto è stato premuto e X è il valore ASCII corrispondente al tasto premuto con il bit 7 definito = (1).
L'utilità di questo comando è notevole nelle lunghe elaborazioni dove il computer accerta se deve interrompere per l'introduzione di nuovi dati, senza bloccare l'esecuzione del programma.

280 POKE - 16368,0
Questo comando deve essere eseguito immediatamente dopo la lettura della tastiera con X = PEEK (-16384), poichè esso ripristina l'attivazione della tastiera consentendo la lettura di un successivo carattere.

MESSAGGI DI ERRORE

Quando viene commesso un qualsiasi errore, sia in MODO DIRETTO che PROGRAMMATO, questo viene presentato sullo schermo come dagli esempi seguenti:

- In MODO DIRETTO visualizza ? TT ERROR
- In MODO PROGRAMMATO visualizza ? TT ERROR IN NL

dove «TT» è il nome dell'errore e «NL» il numero di linea in cui si è verificato l'errore. Segue un elenco dei possibili MESSAGGI DI ERRORE.

SINTAX ERROR = (errore di sintassi)
Mancanza di parentesi in un'espressione, caratteri illegali in una linea, punteggiatura errata ecc..

TYPE MISMATCH = (errore di battitura)
La parte sinistra di una istruzione di assegnazione era una variabile numerica e il lato destro era una stringa o viceversa; oppure una funzione che prevedeva un argomento a stringa ne ha ricevuto uno numerico e viceversa.

BAD SUBSCRIPT = (subscritto errato)
Si è cercato di fare riferimento ad un elemento della matrice che è al di fuori della dimensione della matrice.

ILLEGAL QUANTITY = (quantità illecita)
Il parametro passato ad una funzione matematica oppure ad una stringa era oltre la portata prevista.La causa di questo tipo di errore può essere dovuta a:
1) Un indice di matrice negativo come LET B (-2) = Ø
2) Un uso di SQR con argomento negativo come SQR (-6)
3) Un argomento negativo o uguale a Ø per il logaritmo (LOG) come LOG (-3); LOG (Ø)
4) Elevazione a potenza X^Y con X negativo e Y che non sia un intero
5) Uso scorretto nell'argomento delle funzioni PEEK, POKE, LEFT$, RIGHT$, MID$, ON...GOTO, TAB, SPC, WAIT oppure nelle funzioni per grafici.

ILLEGAL DIRECT = (istruzione illecita)
Si cade in questo errore quando si tenta di usare in MODO DIRETTO i seguenti comandi: INPUT, GET, DATA, DEFFN.

DIVISION BY ZERO = (divisione per zero)
Dividere un numero per zero è considerato un errore.

NEXT WITHOUT FOR = (Next senza FOR)
La variabile di una linea di programma, in una istruzione NEXT non corrisponde alla variabile dell'istruzione FOR eseguita precedentemente.
Deve essere inserita nella linea appropriata, la parte FOR di un ciclo FOR....NEXT oppure cancellare la parte NEXT errata del ciclo.

CAN'T CONTINUE = (non posso continuare)
L'errore è causato dal tentativo di continuare un programma inesistente, oppure dopo la cancellazione o l'aggiunta di una linea in un programma oppure dopo ad un arresto per errore.

UNDEF'D STATEMENT = (istruzione non definita)
Si cade in questo errore quando si tenta di eseguire un comando GOTO, GOSUB oppure THEN ad un numero di linea inesistente.

UNDEF'D FUNCTION = (funzione non definita)
L'errore è causato quando si fa riferimento ad una funzione definita dall'utente che non è mai stata definita.

OVERFLOW = (superamento di capacità della memoria)
Si cade in questo errore quando il risultato di un calcolo è troppo grande per essere rappresentato dalla capacità numerica del BASIC.
Se il risultato che provoca l'errore è negativo, esso viene indicato come «Ø» e l'esecuzione prosegue senza visualizzare nessun messaggio di errore.

FORMULA TOO COMPLEX = (formula troppo complessa)
L'errore è causato se si eseguono più di due istruzioni IF «AA» THEN.

OUT OF MEMORY = (mancanza di memoria)
L'errore è provocato da una delle seguenti cause:

- Programma troppo lungo - espressione troppo complicata - Definizione di LOMEM troppo alta - definizione di LOMEM troppo bassa - Definizione di LOMEM minore del valore corrente - esecuzione di cicli FOR....NEXT nidificati oltre 1Ø livelli - esecuzione di parentesi nidificate oltre 36 livelli - esecuzione di GOSUB nidificati oltre 24 livelli.

OUT OF DATA = (mancanza di dati)
L'errore si verifica quando è stata eseguita una istruzione READ e tutte le istruzioni DATA del programma sono già state lette; oppure il programma ha cercato di leggere troppi dati o i dati inseriti nel programma erano insufficienti.

REDIM'D ARRAY = (matrice ridimensionata)
L'errore si verifica quando viene incontrata una istruzione di dimensionamento (DIM) dopo che ne era stata eseguita una in precedenza.

STRING TOO LONG = (stringa troppo lunga)
Si cade in questo errore quando si tenta di eseguire una stringa di lunghezza superiore a 255 caratteri servendosi dell'operatore di concatenazione (+).

RETURN WITHOUT GOSUB = (ritorno senza GOSUB)
L'errore si verifica quando viene incontrata una istruzione RETURN senza che sia stata eseguita precedentemente l'istruzione GOSUB.

AVVERTENZE PER UN MIGLIORE UTILIZZO DELLA MEMORIA

Le seguenti avvertenze possono essere utili per creare programmi più brevi e risparmiare spazio di memoria.

1 - Usare più istruzioni per ogni linea di programma fino a un massimo di 255 caratteri. C'è un piccolo spazio (5 bytes) associato ad ogni linea nel programma, 2 di questi 5 bytes contengono il numero di linea in forma binaria. Ciò significa che qualsiasi sia il Vostro numero di linea (minimo Ø e massimo 65529) richiederà sempre 2 bytes.
2 - Eliminare tutte le istruzioni REM, poichè essa usa almeno un byte in più del numero di bytes del testo di commento.
3 - Usare variabili invece delle costanti è probabilmente la più importante delle avvertenze, facendo aumentare la velocità di esecuzione di un fattore 1Ø; infatti se in un programma viene usata la costante 3.14159 per 1Ø volte e venga invece inserita nel programma la linea 2ØØ P = 3.14159 e tutte le volte che serve usare la variabile P si risparmieranno circa 4Ø bytes, accellerando anche i tempi di esecuzione.
Utilizzare le stesse variabili temporanee quando si chiede all'utente di dare una risposta (SI o NO) a due differenti domande in due volte diverse durante l'esecuzione del programma.
4 - Il programma può terminare senza l'istruzione END.
5 - Usare le istruzioni GOSUB per eseguire parti di programma che compiono le stesse azioni.
6 - Usare anche gli elementi zero delle matrici per esempio M (Ø), K (Ø,X).

DATI RELATIVI ALL'OCCUPAZIONE DI MEMORIA DI ALCUNE ISTRUZIONI E VARIABILI.

- Ogni carattere usa	1 byte
- Numero di linea usa	4 bytes
- Comandi BASIC tipo FOR, GOTO, NOT, COS, INT, STR $ ecc.	usano 1 byte
- Variabili reali A, intere A% e a stringa A$	usano 7 bytes
- Matrici reali	usano 12 bytes di cui 2 bytes per il nome della variabile, 2 per lo spazio della matrice, 1 per la quantità delle dimensioni, 2 per specificare ogni dimensione e 5 bytes per ogni elemento della matrice.
- Matrici intere	usano 2 bytes per ogni elemento.
- Matrici a stringa	usano 3 bytes per ogni elemento.

- Ogni parentesi usata in un'espressione usa 4 bytes e ogni risultato provvisorio usa 12 bytes.
- Ogni ciclo attivo FOR...NEXT usa 16 bytes.
- Ogni GOBUS attivo (che non sia ancora terminato con un RETURN) usa 6 bytes.

PAROLE RISERVATE AL BASIC

E RELATIVI CODICI DECIMALI

PAROLA CHIAVE RISERVATA	CODICE DECIMALE	PAROLA CHIAVE RISERVATA	CODICE DECIMALE	PAROLA CHIAVE RISERVATA	CODICE DECIMALE
ABS	212	IN #	139	ROT=	152
AND	2Ø5	INPUT	132	RUN	172
ASC	23Ø	INT	211	SAVE	183
AT	197	INVERSE	158	SCALE=	153
ATN	225	LEFT$	232	SCRN(	215
CALL	14Ø	LEN	227	SGN	21Ø
CHR$	231	LET	17Ø	SHLOAD	154
CLEAR	189	LIST	188	SIN	223
COLOR =	16Ø	LOAD	182	SPC(	195
CONT	187	LOG	22Ø	SPEED=	169
COS	222	LOMEM:	164	SQR	218
DATA	131	MID$	234	STEP	199
DEF	184	NEW	191	STOP	179
DEL	133	NEXT	13Ø	STORE	168
DIM	134	NORMAL	157	STR$	228
DRAW	148	NOT	198	TAB(	192
END	128	NOTRACE	156	TAN	224
EXP	221	ON	18Ø	TEXT	137
&	175	ONERR	165	THEN	196
FLASH	159	OR	2Ø6	TO	193
FN	194	PDL	216	TRACE	155
FOR	129	PEEK	226	USR	213
FRE	214	PLOT	141	VAL	229
GET	19Ø	POKE	185	VLIN	143
GOSUB	176	POP	161	VTAB	162
GOTO	171	POS	217	WAIT	181
GR	136	PRINT	186	XDRAW	149
HCOLOR=	146	PR #	138	+	2ØØ
HGR	145	READ	135	-	2Ø1
HGR2	144	RECALL	167	*	2Ø2
HIMEM:	163	REM	178	/	2Ø3
HLIN	142	RESTORE	174	^	2Ø4
HOME	151	RESUME	166	>	2Ø7
HPLOT	147	RETURN	177	=	2Ø8
HTAB	15Ø	RIGTH$	233	<	2Ø9
IF	173	RND	219		

Queste parole, riservate al BASIC, usano solamente 1 byte della memoria.

COME CONVERTIRE PROGRAMMI BASIC NON SCRITTI PER IL LEMON II

Malgrado la somiglianza dei diversi dialetti BASIC che si incontrano sui vari computers, tra essi ci sono però delle incompatibilità che occorre modificare se si vuole far girare lo stesso programma.

1) Alcuni BASIC usano le parentesi quadre, aperta e chiusa, «[» e «]» per indicare gli indici delle matrici. Il LEMON II usa invece le parentesi rotonde, aperta e chiusa «(» e «)».
2) Diversi BASIC obbligano a dimensionare la lunghezza della stringa prima di utilizzarla. Occorre eliminare tutti gli ordini di questo tipo. Una dichiarazione della forma DIM B$ (I, J) indica una matrice stringa di J elementi ognuno dei quali di lunghezza I. Convertire questi ordini DIM negli equivalenti per il LEMON II: DIM B$ (J).
   Come segno di concatenazione di stringhe, nei diversi BASIC si usano is egni «&» oppure «,». Per il LEMON II invece viene usato il segno «+». Inoltre si usa LEFT$, RIGHT$ e MID$ per prelevare le sottostringhe delle stringhe. Alcuni BASIC usano B$ (I) per accedere al I-esimo carattere della stringa B$ e B$ (I, J) per estrarre la sottostringa B$ dalla posizione del carattere I alla posizione del carattere J. Convertirli come segue:

DIALETTI BASIC	**BASIC LEMON II**
B$ (I)	MID$ (B$, I, 1)
B$ (I, J)	MID$ (B$, I, J -I, +1)

   Ciò premette che il riferimento ad una sottostringa di B$ sia in una espressione o sia sul lato destro di una assegnazione. Se il riferimento a B$ è nella parte sinistra di una assegnazione ed X è l'espressione stringa usata per rimpiazzare i caratteri in B$, convertire come segue:

DIALETTI BASIC	**BASIC LEMON II**
B$ (I) = X$	B$ = LEFT$ (B$, I, 1)+X$+MID$ (B$, I,+1)
B$ (I, J) = X$	B$ = LEFT$ (B$, I, 1)+X$+MID$ (B$, J, +1)

3) Alcuni tipi di BASIC consentono di utilizzare assegnazioni multiple come ad esempio: 1ØØ LET A = B = Ø.
   Le variabili A e B vengono poste uguali a Ø. Nel LEMON II si ha un risultato diverso. Tutti i segni di uguale «=» alla destra del primo sono interpretati come operatori logici di confronto. Ciò pone la variabile A a 1 se B fosse uguale a Ø.
   Se B non fosse uguale a Ø, A sarebbe Ø. Un modo corretto per il LEMON II è scriver-

lo come segue: 1ØØ LET B = Ø : A = B.

4) Nei programmi in cui viene usata la funzione MAT, andrà riscritta usando il ciclo FOR....NEXT.
5) Per separare diverse istruzioni in una stessa linea viene usata in alcuni BASIC la «/» al posto del «:». Per il LEMON II occorre sostituire ciascuna «/» con dei «:».

CODICE DEI CARATTERI ASCII

DEC = codice ASCII decimale; HEX = codice ASCII esadecimale; CHAR = nome caratteri ASCII; n/a = non accessibile alla tastiera.

DEC	HEX	CHAR DA BATTERE	CARATTERE
Ø	ØØ	NULL	CTRL @
1	Ø1	SOH	» A
2	Ø2	STX	» B
3	Ø3	ETX	» C
4	Ø4	ET	» D
5	Ø5	ENQ	» E
6	Ø6	ACK	» F
7	Ø7	BEL	» G
8	Ø8	BS	» H o ←
9	Ø9	HT	» I
1Ø	ØA	LF	» J
11	ØB	VT	» K
12	ØC	FF	» L
13	ØD	CR	» M o RETURN
14	ØE	SO	» N
15	ØF	SI	» O
16	1Ø	DLE	» P
17	11	DC1	» Q
18	12	DC2	» R
19	13	DC3	» S
2Ø	14	DC4	» T
21	15	NAK	» U o →
22	16	SYN	» V
23	17	ETB	» W
24	18	CAN	» X
25	19	EM	» Y
26	1A	SUB	» Z
27	1B	ESCAPE	ESC
28	1C	FS	n/a
29	1D	GS	CTRL SHIFT M
3Ø	1E	RS	CTRL ^
31	1F	US	n/a
32	2Ø	SPACE	SPAZIO
33	21	!	!
34	22	»	»
35	23	#	#
36	24	$	$
37	25	%	%
38	26	&	&

39	27	'	'
4Ø	28	(	(
41	29	)	)
42	2A	*	*
43	2B	+	+
44	2C	,	,
45	2D	-	-
46	2E	.	.
47	2F	/	/
48	3Ø	Ø	Ø
49	31	1	1
5Ø	32	2	2
51	33	3	3
52	34	4	4
53	35	5	5
54	36	6	6
55	37	[illegible]	7
56	38	8	8
57	39	9	9
58	3A	:	
59	3B	;	;
6Ø	3C	<	<
61	3D	=	=
62	3E	>	>
63	3F	?	?
64	4Ø	@	@
65	41	A	A
66	42	B	B
67	43	C	C
68	44	D	D
69	45	E	E
7Ø	46	F	F
71	47	G	G
72	48	H	H
73	49	I	I
74	4A	J	J
75	4B	K	K
76	4C	L	L
77	4D	M	M
78	4E	N	N
79	4F	O	O
8Ø	5Ø	P	P
81	51	Q	Q
82	52	R	R
83	53	S	S
84	54	T	T
85	55	U	U
86	56	V	V

87	57	W	W
88	58	X	X
89	59	Y	Y
9Ø	5A	Z	Z
91	5B	[	n/a
92	5C	/	n/a
93	5D	]	]
94	5E		
95	5F	-	n/a

Dal 96 al 255 si ottiene su LEMON II la ripetizione dei caratteri generati da Ø a 95. Comunque sia CHR$ (65) che CHR$ (193) generino una A, non vengono riconosciuti come caratteri uguali, quando si usano delle stringhe; anche una stampante collegata al LEMON II li stamperà diversamente.

MAPPA DI MEMORIA

INDIRIZZO DI MEMORIA		DESCRIZIONE
HEX	DECIMALE	
Ø ÷ 1FF	Ø ÷ 511	Spazio riservato al Sistema non disponibile per l'utente.
2ØØ ÷ 2FF	512 ÷ 767	Buffer dei caratteri della tastiera.
3ØØ ÷ 3FF	768 ÷ 1Ø23	Spazio disponibile per l'utente.
4ØØ ÷ 7FF	1Ø24 ÷ 2Ø47	Spazio per il testo di pagina 1 o per i grafici a colori.
8ØØ ÷ BFFF	2Ø48 ÷ 49151	Spazio disponibile per i programmi d'utente. Corrisponde alla memoria RAM totale o meno se vengono riservate delle aree per routines in linguaggio macchina e per grafici ad Alta risoluzione.
2ØØØ ÷ 3FFF	8192 ÷ 16383	Spazio per la pagina 1 dei grafici ad ALTA RISOLUZIONE.
4ØØØ ÷ 5FFF	16384÷24575	Spazio per la pagina 2 dei grafici ad ALTA RISOLUZIONE.
CØØØ ÷ CFFF	49152÷53247	Spazio per gli indirizzi I/O hardware.
DØØØ ÷ DFFF	53248÷57343	Spazio riservato a futura espansione ROM.
EØØØ ÷ F7FF	57344÷63487	Spazio occupato dall'interprete BASIC.
F8ØØ ÷ FFFF	63488÷65535	Spazio occupato dal programma Monitor del LEMON II.